Amendments to the Specification:

Please replace the title with the following rewritten title:
"METHOD OF DETERMINING THE ROUTING OF INTERCONNECTED REGIONS"

Please replace the paragraph beginning on page 1, line 9 through page 1, line 20 with the following rewritten paragraph:

Various computer programs (so-called "routers") are known for the routing of tracks of electrical circuits. These are so-called "serial" routers. In such routing, a "connection" indicates which regions of the circuit must be connected together. A serial router considers each requested connection separately and realises it into conductive track segments before moving on to the next connection. There are many cases where a standard serial router will never find the routing solution that an experienced human router might find. Consider the connections A, B, C in Figure 1(a). A serial router will always route one of these first, and will then route the others round it. If the router starts with connection A, it may produce a solution rather like Figure 1(b). It can never find the optimal solution, shown in Figure 1(c), unless by some non-serial part of its algorithm, e.g. a post-processing tool. An experienced human router would instantly find the solution of Figure 1(c) by seeing the problem as a whole.

Please replace the paragraph beginning on page 4, line 15 through page 5, line 20 with the following rewritten paragraph:

There will now be described with reference to Figure 2 an algorithm of the example.

(i) Split space into regions 20. This algorithm already existed in an equispace routing tool of Zuken Limited, 1500 Aztec West, Almondsbury, Bristol BS32 4RF, UK and is a sophisticated extension of the Delaunay triangulation algorithm. In Zuken's implementation, the regions are polygons and the shared boundaries are edges. Figure 3 shows an example board 30 and the resulting polygonalisation 31.

(ii) Route each connection 21. This step is a fine-tuned flood search algorithm, stepping from one polygon to the next, attempting to find a permissible route for each connection considered in isolation.

(iii) Consult shared boundaries 22. Part of the concurrency comes from each boundary's ability to examine all connections across it and discourage the least promising (by making them less probable, or more expensive) if there are too many. A shared boundary also decides the crossing order for its connections.

(iv) Consult regions 23. Regions are also consulted, allowing them to place putative vias, veto crossing tracks, and so on. They also help assemble conflict objects.

(v) Build conflicts 24. Since connections are routed independently, respecting only conflicts they have been explicitly told to respect, there is likely to be crossing routing. A conflict object holds information about a region of the board and the paths which cross there, and these are built with help from regions, shared boundaries, and the connections themselves.

(vi) Resolve conflicts 25. These conflicts are then resolved independently as far as possible. Often they cannot be resolved fully, so they are partially resolved ready for the next pass which builds lighter conflicts (which are generally easier to deal with).

(vii) Realise tracks 26. Once the homotopy classes of all connections are known, they can be converted to real conducting tracks and vias by existing Zuken tools. If the conflict resolution loop has been executed more than a fixed number of times, or if passes round this loop are no longer adding to the routing, as many tracks as possible are realised, and the algorithm ends.

Please replace the paragraph beginning on page 5, line 21 through page 5, line 24 with the following rewritten paragraph:

The following is an example of a routing problem as solved by the method. For simplicity, the example consists of a small region of a single-layer design with only "2-base" (i.e.

source-to-target) connections. The input is as in Figure 4, required connections being indicated by dotted lines with the required connections being pin 1 to pin A, pin 2 to pin B, pin 3 to pin C, pin 4 to pin D, and pin 5 to pin E.